An Experiment on Mizar Adjectives with Visible Arguments

Adam Naumowicz

Institute of Informatics
University of Białystok, Poland
adamn@mizar.org

SYNASC 2020, September 1, 2020

Outline of the talk

- Adjectives in MIZAR
  - absolute
  - with arguments
  - with extra visible (implicit) arguments
- Objectives and results of a case study
- Conclusions and plans for future work

Adjectives in Mizar

- Introduced to the Mizar language around 1983/84
- Expressing formal statements in a much more natural way than just using a sort of first order logic notation
- Evolved from simple "syntactic sugar" to a powerful automation mechanism:
  - necessary for disambiguation based on a (soft) type system
  - intensively used by the inference checking engine

Adjective arguments

- Example first order formula: `Number(X) & NOT Prime(X)`
- MIZAR rendering: `X is non prime number`
  - `number` is a notion of type
  - `prime` (as well as `non prime`) is an adjective applicable to this type
- Every occurrence of an adjective always has one (visible) argument – its subject (in this case `X`)
  - For *absolute* adjectives this is the only argument
- In general, adjectives can have a number of hidden arguments (inherited from the type of `X`)
- Extra visible arguments: `X is n-dimensional`
  - `X` is the `-dimensional` adjective's standard argument `n` is applied as an extra visible argument

The objectives of this case study

- Check if (theoretically possible) replacing old adjectives left in the library by corresponding generalized notions utilizing the new feature is feasible
- Focus on adjectives with extra visible arguments related to number sets
  - High priority for MML integrity
  - Long term process already started back in 2009
- Implement an enhancement of the MIZAR system to proceed with refactoring selected MML theories

MIZAR adjective round-up algorithm

- Calculate the soft type specification of every term in an inference
  - The soft type becomes a list of types accompanied by clusters of applicable adjectives computed as a logical consequence of imported registrations applied to the initial user-provided list of adjectives
  - Processing the type data in the inference checking can utilize premises collected from all members of classes of equal terms
  - Syntactic identification can only rely on the type information present in a particular statement
- This information is augmented by available re-definitions of underlying notions and consequences of user-provided adjectives imported with conditional adjective registrations (so called rounded-up clusters of adjectives).
- Example: an object introduced in a piece of reasoning as a **real-valued** function can be used wherever the context requires a **complex-valued** function

Example conditional registration

```
registration
  cluster real-valued -> complex-valued
        for Relation;
end;
```

Using the syntax of adjectives with visible arguments, the same conditional information is encoded in the same article as:

```
registration
  cluster REAL-valued -> COMPLEX-valued
        for Relation;
end;
```

but this time a more general `-valued` adjective is used together with two constants `REAL` and `COMPLEX` representing the MML definitions of the sets of real and complex numbers, respectively.

MML refactoring goal 1

- Get rid of the definitions:

```
definition
  let f be Relation;
  attr f is complex-valued means
        rng f c= COMPLEX;
  attr f is ext-real-valued means
        rng f c= ExtREAL;
  attr f is real-valued means
        rng f c= REAL;
  attr f is natural-valued means
        rng f c= NAT;
end;
```

- Replace with:

```
definition
  let X be set, R be Relation;
  attr R is X-valued means rng R c= X;
end;
```

Occurrences of *-valued adjectives

*-valued **adjective**	**Occurrences**
complex-valued	603
COMPLEX-valued	22
ext-real-valued	81
ExtREAL-valued	6
real-valued	847
REAL-valued	43
RAT-valued	93
INT-valued	338
natural-valued	337
NAT-valued	52

MML refactoring goal 2

```
definition
  let X be set;
  attr X is complex-membered means
        x in X implies x is complex;
  attr X is ext-real-membered means
        x in X implies x is ext-real;
  attr X is real-membered means
        x in X implies x is real;
  attr X is rational-membered means
        x in X implies x is rational;
  attr X is integer-membered means
        x in X implies x is integer;
  attr X is natural-membered means
        x in X implies x is natural;
end;
```

- Replace with

```
definition
  let X,Y be set;
  attr Y is X-membered means x in Y implies x in X;
end;
```

Occurrences of *-membered adjectives

*-membered **adjective**	**Occurrences**
complex-membered	162
ext-real-membered	301
real-membered	348
rational-membered	104
integer-membered	108
natural-membered	174

MML refactoring procedure

- Most of the work required syntactic replacements which can easily be automatized
- There were cases where the replaced definitions interfered with library import mechanisms
  - The problem was usually solved by adjusting the list of imported constructors, the order of notations or changed references to theorems
  - Definitional expansions previously referring to the article `VALUED_0` should be replaced by references to `RELAT_1`

MML refactoring caveats

- An interesting case emerged with the article MESFUNC2, where authors decided to re-define the notion of a real-valued function:

```
definition
  let X, f;
  redefine attr f is real-valued means
        for x st x in dom f holds
                  |. f.x .| < +infty;
end;
```

- This was a typical registration vs. redefinition problem
- The version with no argument could be redefined this way, whereas a general notion of the -valued adjective with an extra constant (REAL) cannot
- The solution was to replace the redefinition with a corresponding theorem statement
- The redefinition was used only in the article which introduced it, so the impact was negligible

MML refactoring caveats – ctd.

- In another article of the same series (MESFUNC7) the authors created a synonym:

```
notation
  let F be Relation;
  synonym F is extreal-yielding for
  F is ext-real-valued;
end;
```

- That could not be done for an adjective with a given argument, so the new notion should be dropped

Conclusion and future work

- The experiment has proved that the application of the more general mechanism of adjectives with visible arguments is feasible
- Users should be encouraged to make use of the new way of defining adjectives, taking into consideration the possible caveats
- The question for the future work: should we enforce the implementation of the enhanced adjective round-up algorithm into the mainstream Mizar tools?
  - It seems promising, provided there is a mechanism developed (on the language side) to specify a concrete instance of an adjective to be used for identification
  - We need a stronger "qua" (type qualification), or rather a more general mechanism with better syntax to allow users to disambiguate the type specification